

A bridge between logic and neural

甄景贤 (King-Yin Yan)

General.Intelligence@Gmail.com

Abstract. Logic-based AI and connectionist AI has long been separated, but we found a correspondence between the two sides. Logic structure is similar to human natural language, whereas the brain is connectionist. A chief goal of machine learning is to use inductive bias to speed up learning, but this goal may seem a bit aimless without guidance. If we add logical structure onto neural structure, we give it more constraints, ie. inductive bias.

On the logical side, the structure is highly abstract and symbolic, but learning is too slow; My aim is to build a “bridge” to **transfer** logical structure to the neural side.

This problem has taken a long time to solve because the structure of logic is not a common mathematical structure; it is difficult merely to articulate this structure in mathematical terms. That is until I discovered the use of model theory, that I finally got a satisfactory solution:

I will first explain the structure of the logic side, and then the structure of the neural side.

1 Structure of logic

A **logic system** can be defined as:

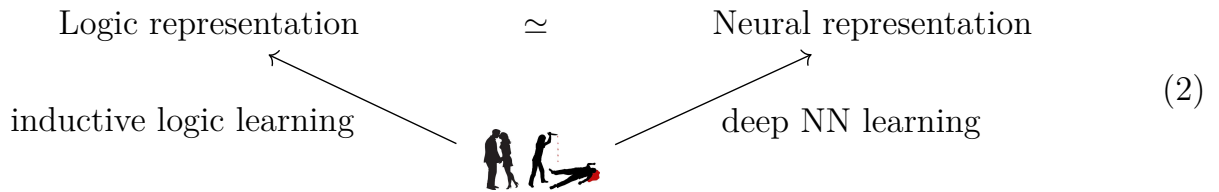
- a set of symbols for **constants**, **predicates**, and **functions**
- **propositions** built from the above atoms
- **connectives** between simple propositions, eg: \neg, \wedge, \vee
- The **logical consequence** relation: $\Gamma \vdash \Delta$

Personally I think **relation algebra** [12] [8] is closer to human natural language, but the standard form used in mathematical logic research is first-order logic (FOL). This is however

not of the essence, because all logics are basically easily interconvertable. In the following we concentrate on FOL.

From raw sensory data, we can **inductively** learn a set of logic formulas, ie. the knowledge base $\boxed{\text{KB}}$. This process is under the research heading of **inductive logic programming** (ILP). Everyone familiar with classical AI knows ILP, but in recent decades the popular focus is on **statistical learning**, and this kind of symbolic logic learning is relatively neglected.

Raw sensory data can be processed using neural-based pattern recognition, or it can be processed via ILP-based pattern recognition. The result of these two pathways should obviously be isomorphic (at least approximately):



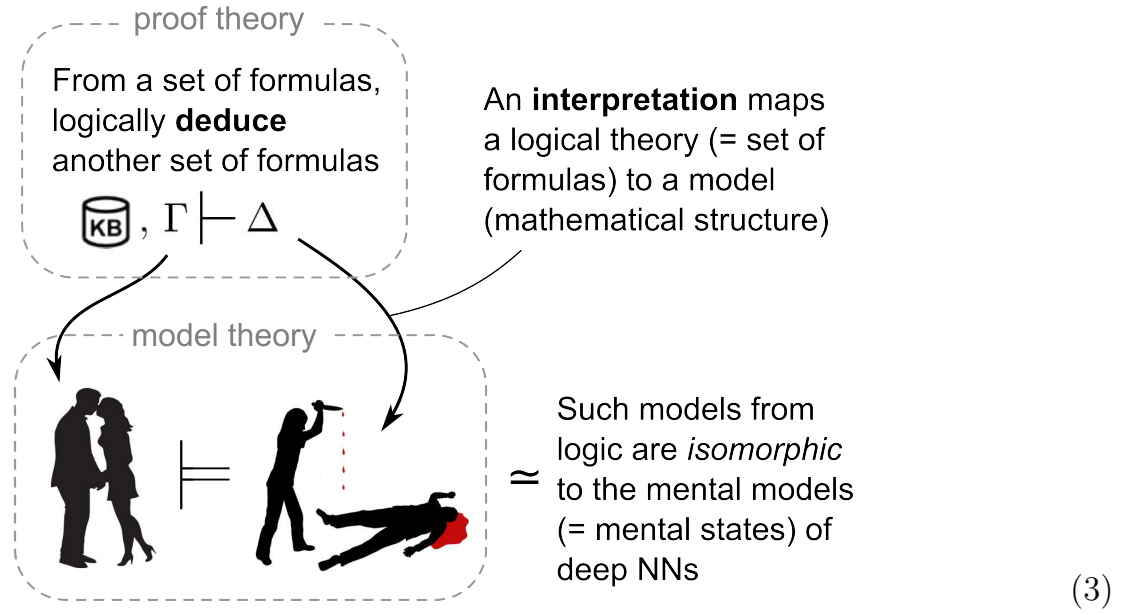
I have spent a lot of time thinking how to transition logic-based representations to the neural side, but found this target to be very elusive.

On the one hand, logic is the study of the laws of human thinking, developed over centuries. The logical description is correct; There should be a correspondence between logic and neural, as they are both doing the same thing (intelligence).

In **cognitive science**, many researchers believe the representations inside our brains are so-called “mental models”, and few would believe that the brain uses a representation like the propositions in symbolic logic, or to think with the sort of symbolic manipulations as λ -calculus.

As an example, when we verbally describe a case of murder, the reader would establish in her mind’s eye a “model” that is similar to real experience yet is unreal. The human brain seems to use such mental models to think, rather than with a bunch of propositions. Examples of model-based reasoning using logic are [10] [9].

So eventually I realized that the logic-neural correspondence must be achieved through model theory:



\vdash means deducing from a set of (symbolic logic) **formulas** to new formulas. \models means that a **model** necessarily entails another model.

2 Model theory

For the basics of model theory, cf [4] [11]. The model-theoretic approach is to separate the **symbolic language** \mathcal{L} and the \mathcal{L} -structures it refers to. They are related by **interpretation** maps.

\mathcal{L} contains the set of symbols (predicates, relations, functions, constants), which recursively generates formulas and compound formulas. These are the **symbolic** stuff.

An \mathcal{L} -structure can be any abstract algebraic structure. It usually consists of a **base set**, and functions and relations defined over that set.

The central idea of model theory is using the interpretation map i to “preserve” certain relations, for example:

$$R(a, b) \xrightarrow{i} R^{\mathcal{M}}(a^{\mathcal{M}}, b^{\mathcal{M}}) \quad (4)$$

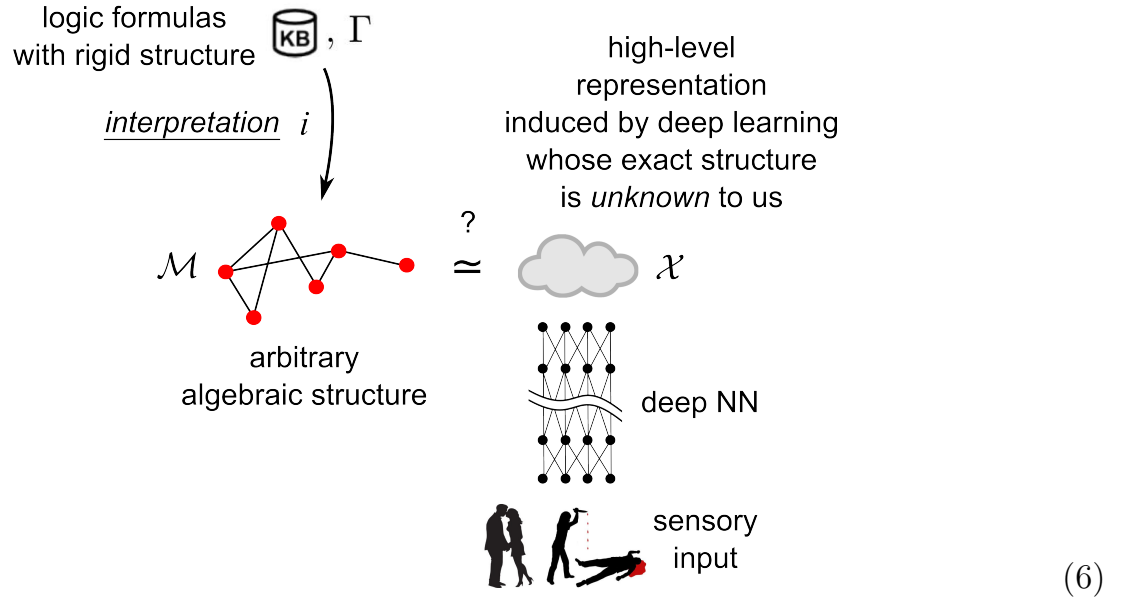
R is a relation, $x^{\mathcal{M}}$ denotes the object corresponding to x in the structure \mathcal{M} . The left side is symbolic logic; the right side is the actual structure. When model theory is applied to first-order logic, we get the equivalence of \vdash and \models (which looks like tautology). This can be found in any mathematical logic textbook, such as [6].

If we use category theory to express the correspondence between logic and neural:

$$\begin{array}{ccc}
 \mathcal{L} & & \\
 \downarrow i & & \\
 \mathcal{M} & \simeq & \mathcal{X} \\
 & & \uparrow \text{deep NN} \\
 & & \mathcal{S}
 \end{array} \tag{5}$$

- \mathcal{L} = category of logic theories (= sets of formulas)
- i = interpretation maps
- \mathcal{M} = category of models (from logic)
- \mathcal{X} = category of models (from deep NNs)
- \mathcal{S} = sensory input

The above diagram says the same as this cartoon explanation:




In other words, $\mathcal{X} = \text{cloud}$ is **induced** from deep learning; but its structure is **opaque** to us (that is the weakness of neural networks).

Whereas, the structure of $\mathcal{M} = \text{graph}$ is the subject matter of model theory.

In model theory, \mathcal{L} is the category of logic formulas, $\mathcal{M} = \text{graph}$ could be any algebraic structure. We just need to map the constants, predicates, relations, functions in \mathcal{L} to the structure in \mathcal{M} . For the sake of conciseness, we will just constants symbols and relation symbols, as these two are the **essence** of logic.

$$\begin{array}{ccc}
 \mathcal{L} & \xrightarrow{i} & \mathcal{M} \\
 \text{constant symbol} & \mapsto & \bullet \\
 \text{relation symbol} & \mapsto & \bullet \text{---} \bullet
 \end{array} \tag{7}$$

Our problem is that we don't know the structure of . For a long time, researchers have treated neural networks as “black boxes”, but if we don't know the structure of $\mathcal{X} = \text{cloud}$ we cannot build the isomorphism $\mathcal{M} \simeq \mathcal{X}$.

3 The structure of neural networks

So, what is the structure of neural network representations?

Basically a **neural network** is:

$$F(\mathbf{x}) = \textcircled{S}(W_1 \textcircled{S}(W_2 \dots \textcircled{S}(W_L \mathbf{x}))) \quad (8)$$

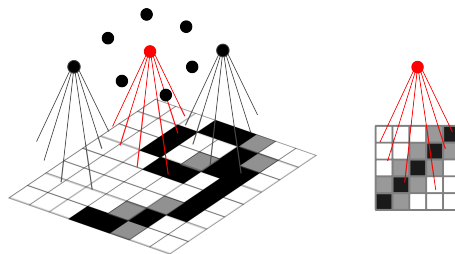
where L is the number of layers, W_ℓ is the **weight matrix** for each layer, \textcircled{S} is the sigmoid function applied component-wise to each neuron (whose role is to provide **non-linearity**).

\textcircled{S} acts on each component of \mathbf{x} ; Its action is **not invariant** under a change of coordinates. Therefore, \textcircled{S} is not a vector operation, and thus \mathcal{X} is also not a **vector space** structure. Common practice is to denote \vec{x} as a vector, but this is misleading.

If we connect a neural network **head to tail** forming a loop, we get the simplest form of a cognitive architecture, whose **state equation** is $\mathbf{x}_{n+1} = F(\mathbf{x}_n)$, or in continuous time $\dot{\mathbf{x}} = f(\mathbf{x})$. From this we can see that $\mathbf{x} \in \mathcal{X}$ is a **differential manifold**. The deeper theory is that it is a Hamiltonian system, having the **symplectic** structure. In other words, \mathcal{X} is a differential manifold endowed with a symplectic metric. But this is outside the scope of this paper, please refer to my [16].

Now let's think about how a neural network performs pattern recognition, perhaps it would be illuminating:

Consider the simplest case, eg. a layer of neural network extracting visual features from the digit “9”. This layer may have many neurons (left figure), each neuron locally covers a region of the input layer, the so-called “local receptive field” in the neuroscience of vision (right figure).

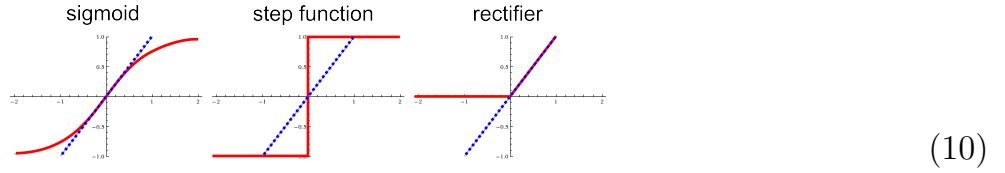


(9)

Suppose the red neuron is responsible for recognizing the “diagonal line” feature. Its equation is $\mathbf{y} = \textcircled{S}(W\mathbf{x})$. The matrix W 's role is to affine “rotate” the feature space, so that the features we desire is pointing in a certain direction. Then we use \textcircled{S} to “squeeze” the desired and undesired features. The output after \textcircled{S} represents the **presence or absence** of a particular feature, ie $\{0, 1\}$. This is a form of information **compression**.

Let's digress a bit on **chaos theory**: the role of \textcircled{S}^{-1} is to “stretch”, dragging points that are close neighbors to more distant positions. Looking at the common **activation functions**,

we see they are all non-linear **deformations** away from the identity $y = x$:



This is very similar to the “horseshoe” proposed by Steven Smale [13], a recipe for creating chaos. In other words, “stretching” and then putting back to the original space, and repeating this process, will create chaos [5] [14]. (The neural network running backwards in time has \mathcal{O}^{-1} , so its forward movement is also chaotic.) A variant of the Smale horseshoe is the “baker map”, analogous to kneading dough in bakery.

To summarize:

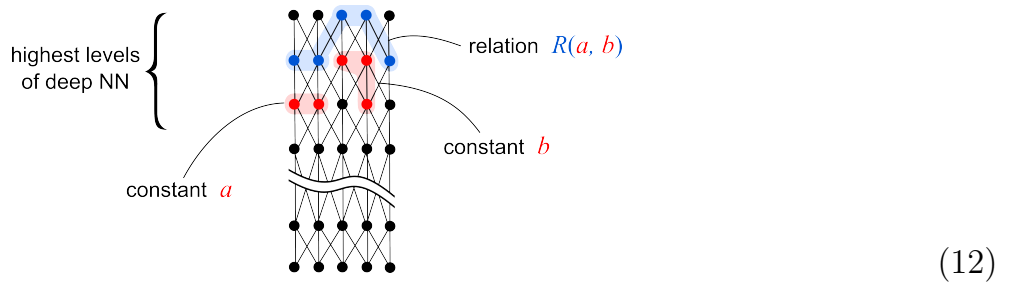
Each neuron represents the presence or absence of a certain **feature**.

Higher-layer neurons represents **relations** between lower-layer features.

Following this line of thinking, we may conjecture this correspondence:

$$\begin{array}{llll}
 \mathcal{M} & \simeq & \mathcal{X} & \\
 \text{constant} & \bullet & \Leftrightarrow & \text{neuron} \\
 \text{relation} & \bullet\text{---}\bullet & \Leftrightarrow & \text{relation between higher and lower neurons}
 \end{array}
 \tag{11}$$

But beware that this correspondence may not be 1-1, it could be one constant corresponding to several neurons’ (linear?) **combination**. The actual situation may be illustrated as follows (each layer may contain a vast number of neurons):



$R(a, b)$ can be searched for among the **common parents** of a, b (eg. those blue neurons; The value of $R(a, b) =$ a certain linear combination of blue neurons). This can be verified by the condition: If the signals of a and b are both “present”, the value of $R(a, b)$ should also be “true”.

4 Conclusion

This paper is not very successful because we have jumped to the conclusion (11) and (12) without rigorous justification: They are merely intuitively plausible. In theory, since we have known how \mathcal{M} is generated, and how \mathcal{X} is generated, it should be feasible to build a “highway” between them. In practice, it seems that we just need a deep neural network, because neural networks are **universal function approximators**, we don’t even need to care about the structure between \mathcal{M} and \mathcal{X} .

For further research, I hope some professional mathematicians can help with these problems:

1. On the logic side, could we transform its structure to one in **algebraic geometry**? In other words: logic formulas \simeq algebraic equations. I only know of one instance of this kind of algebraic logic: Andreka *et al*'s [2], it seems to be an esoteric research area.
2. From the structure of \mathcal{M} and \mathcal{X} , can we find out the simplest form of a bridge between them? Perhaps using mathematical induction, we examine the generative process of \mathcal{M} and \mathcal{X} layer by layer?

Application: When applying deep learning to natural language understanding, this theory may be helpful.

5 Prior art

- Bader, Hitzler, Hölldobler and Witzel proposed in 2007 a method of neural-symbolic integration [3]. First they generate a Herbrand model¹ from the logic theory, maps the Herbrand model to a fractal space, and then directly use a neural network to learn that fractal space. Though they used model theory, they did not make use of the correspondence between \mathcal{M} and \mathcal{X} in this paper.
- Khrennikov, in a series of papers starting from 1997, proposed using p -adic algebra to model the mental space \mathcal{X} , see his book co-authored with Anashin [1]. A p -adic number can be regarded as a “decimal” number in base p , where p is any prime.
- Classical logic is binary; There has been countless attempts to extend it to fuzzy or probabilistic logic (eg. I have attempted in [17]), but there is still not a unified consensus theory. A slightly different approach, is to regard points in a space as first-order objects, and predicates as functions over that space. Then we get a metric structure and on it a continuous first-order logic (CFOL) [15]. This is a possible structure for \mathcal{M} .
- In model theory there are ultra-filters and ultra-products, which originated in functional analysis. Recently there has been novel research crossing between model theory and Banach space [7]. Simply put, the ultra-product is a way to multiply existing models to yield new models. I looked at it briefly and found that ultra-products often involve sets of infinite cardinality and are very “big” constructs. So they may not be very practical for computational implementation.

Acknowledgement

Thanks to Ben Goertzel for discussions on the AGI mailing list. Ben first pointed out the advantage of neural network learning over inductive logic learning, which prompted me to research their relationships. Thanks also to Juan Carlos Pinto for personal discussions on the structure of neural networks.

¹ Herbrand model is a common construct in logic-based AI. The gist is to generate from the logic language \mathcal{L} “whatever that can be instantiated”, resulting in an (often infinite) set of ground sentences (ie, without variables). In other words, Herbrand models are generated from the language \mathcal{L} self-reflexively, without relying on any external structure. Every logic theory has at least one Herbrand model.

References

1. Vladimir Anashin and Andrei Khrennikov. *Applied algebraic dynamics*. de Gruyter, 2009.
2. Andreka, Nemeti, and Sain. *Handbook of philosophical logic*, chapter Algebraic logic, pages 133–247. Springer, 2001.
3. Bader, Hitzler, Hödöbler, and Witzel. The core method: Connectionist model generation for first-order logic programs. *Studies in Computational Intelligence* 77, 205-232, 2007.
4. Kees Doets. *Basic model theory*. CSLI notes, 1996.
5. Robert Gilmore and Marc Lefranc. *The topology of chaos: Alice in stretch and squeezeland*. Wiley-VCH, 2011.
6. Shawn Hedman. *A first course in logic*. Oxford, 2004.
7. José Iovino. *Applications of model theory to functional analysis*. Dover, 2002.
8. Roger Maddux. *Relation algebras*. Elsevier, 2006.
9. Magnani, Nersessian, and Pizzi, editors. *Logical and computational aspects of model-based reasoning*. Kluwer, 2002.
10. Magnani, Nersessian, and Thagard, editors. *Model-based reasoning in scientific Discovery*. Kluwer, 1999.
11. Maria Manzano. *Model theory*. Oxford, 1999.
12. Gunther Schmidt. *Relational mathematics*. Cambridge, 2010.
13. Stephen Smale. Differentiable dynamical systems. *Bulletin of the American Mathematical Society*, 1967.
14. Tamás Tél and Márton Gruiz. *Chaotic dynamics: an Introduction based on classical mechanics*. Cambridge, 2006.
15. Itai Ben Yaacov, Alexander Berenstein, C Ward Henson, and Alexander Usvyatsov. Model theory for metric structures. In *Model theory with applications to algebra and analysis, vol 2*. Cambridge, 2008.
16. King Yin Yan. Wandering in the labyrinth of thinking – a cognitive architecture combining reinforcement learning and deep learning. to be submitted AGI 2017.
17. King-Yin Yan. Fuzzy-probabilistic logic for common sense reasoning. *Artificial general intelligence 5th international conference, LNCS 7716*, 2012.